UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/606,643	06/25/2003	Michael A. Rothman	42P16422	4998

7590 04/19/2007

Cory G. Claassen
BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP
Seventh Floor
12400 Wilshire Boulevard
Los Angeles, CA 90025-1026

EXAMINER	
VUU, HENRY	
ART UNIT	**PAPER NUMBER**
2179	

SHORTENED STATUTORY PERIOD OF RESPONSE	MAIL DATE	DELIVERY MODE
3 MONTHS	04/19/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

If NO period for reply is specified above, the maximum statutory period will apply and will expire 6 MONTHS from the mailing date of this communication.

Office Action Summary	Application No.	Applicant(s)	
	10/606,643	ROTHMAN ET AL.	
	Examiner	Art Unit	
	Henry Vuu	2179	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☐ Responsive to communication(s) filed on 22 January 2007.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1, 2, 4 - 21, 23 - 31* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1, 2, 4 - 21, 23 - 31* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *25 June 2003* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

Applicant's arguments with respect to claims 1, 2, 4 – 21, 23 – 31 have been considered but are moot in view of the new ground(s) of rejection.

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

Claims 12, 13, 15, 17 – 19, and 21 are rejected under 35 U.S.C. 103(a) as being unpatentable over Reasor et al. (Pub. No. 2004/0083196) in view of Crisan et al. (Publication No. 2003/0172372).

As to independent claim 12, Reasor et al. teaches:

A method, comprising: converting hardware configuration settings (see e.g., Fig. 2, para. [0013] and para. [0024]; i.e., the configurable hardware properties stored in RAM are converted to display the XML browser interface of Fig. 2) being stored in firmware of a computing device (see e.g., para. [0013]; i.e., firmware and the data repository is stored in RAM) to a markup language (see e.g., Fig. 2 and para. [0024]; i.e., the hierarchical tree view is represented using XML, and may incorporate Cascading Styling Sheets (CSS), or Extensible Style Sheet Language Transformations (XSLT)); and conveying the markup language to a browser to display the hardware configuration settings in the browser (see e.g., Fig. 2 and para. [0024]), but does not

specifically mention pre-boot runtime of the computing device. Crisan et al. teaches a pre-boot runtime of the computing device (see e.g., para. [0007]; i.e., the "system ROM" is a storage area of information used to initialize the basic input/output system "BIOS" of a computing device, wherein the powering-up of a computing device will execute the low-level functions, such as the BIOS and validity of peripheral devices discussed by Crisan et al., and further initialize the power on self test (POST). POST, is well known to one of ordinary skill in the art for querying and polling peripheral devices during system initialization, prior (emphasis added) to the initialization of the operating system (OS), wherein POST routines test various peripheral devices connected to the computing device in order to properly setup the utilization of the peripheral devices). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the hardware configuration settings being stored in firmware of a computing device of Reasor et al. with the pre-boot runtime of the computing device of Crisan et al. because system ROM associated with the execution of the BIOS and POST of a particular computing system is able to be flashed during system initialization, in which the result of flashing the ROM during every system power-up will allow the system ROM to be constantly up to date (see e.g., para. [0007] and para. [0011]).

As to dependent claim 13, Reasor et al. teaches changing at least one of the hardware configuration settings (see e.g., para. [0023]; i.e., the changing of at least one hardware configuration setting corresponds to adding the property "Size" 400 to one of the hardware's described in Fig. 1) stored in the firmware (see e.g., para. [0013]; i.e., firmware and the data repository is stored in RAM) in response to input received via the browser (browser – see e.g., Fig. 4 and para. [0024]; i.e., Fig. 4 is a markup language compatible browser that allows

hardware property configuration), but does not specifically mention a pre-boot runtime of the computing device. Crisan et al. teaches a pre-boot runtime of the computing device (see e.g., para. [0007]; i.e., the "system ROM" is a storage area of information used to initialize the basic input/output system "BIOS" of a computing device, wherein the powering-up of a computing device will execute the low-level functions, such as the BIOS and validity of peripheral devices discussed by Crisan et al., and further initialize the power on self test (POST). POST, is well known to one of ordinary skill in the art for querying and polling peripheral devices during system initialization, prior (emphasis added) to the initialization of the operating system (OS), wherein POST routines test various peripheral devices connected to the computing device in order to properly setup the utilization of the peripheral devices). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate changing at least one of the hardware configuration settings stores in firmware of Reasor et al. with the pre-boot runtime of the computing device of Crisan et al. because system ROM associated with the execution of the BIOS and POST of a particular computing system is able to be flashed during system initialization, in which the result of flashing the ROM during every system power-up will allow the system ROM to be constantly up to date (see e.g., para. [0007] and para. [0011]).

As to independent claim 15, Reasor et al. teaches a computer-accessible medium (ROM – see e.g., para. [0012]) that provides instructions (software program – see e.g., para. [0012]) that, if executed by a computing device (computer –see e.g., para. [0012]), will cause the computing device to perform operations comprising: generating a browser page to display hardware configuration settings (see e.g., Fig. 2 and para. [0024], lines 4 – 9; i.e., the browser is capable of

displaying configurable hardware properties) of hardware entities of a computing device (see e.g., para. [0012], lines 14 – 22; i.e., the hardware entities corresponds to peripheral devices that may be externally connected, such as CPU, and memories, in which the system firmware identifies and queries connected hardware devices to determine the hardware properties for storing in the database) using a browser (see e.g., para. [0024], the browser corresponds to an XML compatible browser), the hardware configuration settings based at least in part on data structures (see e.g., para. [0013]; i.e., the configurable properties and data structure of the hardware device corresponds to the configurable hardware device stored in the database) provided by the hardware entities (see e.g., para. [0012], lines 14 – 22; i.e., the hardware entities corresponds to peripheral devices that may be externally connected, such as CPU, and memories); and changing at least one of the hardware configuration settings (see e.g., para. [0023]; i.e., the changing of at least one hardware configuration setting corresponds to adding the property "Size" 400 to one of the hardware's described in Fig. 1) in response to input received via the browser (browser – see e.g., Fig. 2 and para. [0024]; i.e., Fig. 2 is a markup language compatible browser that allows hardware property configuration), but does not specifically mention a pre-boot runtime of the computing device. Crisan et al. teaches a pre-boot runtime of the computing device (see e.g., para. [0007]; i.e., the "system ROM" is a storage area of information used to initialize the basic input/output system "BIOS" of a computing device, wherein the powering-up of a computing device will execute the low-level functions, such as the BIOS and validity of peripheral devices discussed by Crisan et al., and further initialize the power on self test (POST). POST, is well known to one of ordinary skill in the art for querying and polling peripheral devices during system initialization, prior (emphasis added) to the

initialization of the operating system (OS), wherein POST routines test various peripheral devices connected to the computing device in order to properly setup the utilization of the peripheral devices). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate a computer-accessible medium that generates a browser page to display hardware configuration settings of Reasor et al. with the pre-boot runtime of the computing device of Crisan et al. because system ROM associated with the execution of the BIOS and POST of a particular computing system is able to be flashed during system initialization, in which the result of flashing the ROM during every system power-up will allow the system ROM to be constantly up to date (see e.g., para. [0007] and para. [0011]).

As to dependent claim 17, Reasor et al. teaches:

The computer-accessible medium (ROM – see e.g., para. [0012]) of claim 15 wherein the instructions for generating the browser page further include instructions (software program – see e.g., para. [0012]) to display the hardware configuration settings (see e.g., Fig. 2 and para. [0019] – [0020]; i.e., Fig. 2 corresponds to a device tree displayed on a device interface used to configure settings, such as adding a "Size" 400 property shown in Fig. 4) based at least in part on the data structures (see e.g., para. [0019]; i.e., the data structures corresponds to the database that is used to store the hardware properties, which in turn is used to construct the device tree 250) and nonvolatile data associated with the hardware entities (see e.g., "Microsoft Computer Dictionary 5th edition" and para. [0011]; i.e., non-volatile memory is defined as "A storage system that does not lose data when power is removed from it. Intended to refer to core memory, ROM, EPROM…", in which the functionality of hardware configuration comprises system software code stored on read-only-memory (ROM) or solid-state-memory).

As to dependent claim 18, Reasor et al. teaches:

The computer-accessible medium of claim 15 wherein the data structures (see e.g., para. [0013]; i.e., building the central repository and providing data structures corresponds to holding information about the hardware device in RAM as they are discovered, in the form of a tree format using descriptive properties and attributes that can be converted into XML) are described using a language convertible to a markup language (see e.g., para. [0017], lines 16 – 18; i.e., the tree format data structure attributes and properties used to build the central repository is able to be parsed to display a hierarchal tree using an Extensible Markup Language (XML) browser, such as Fig. 2).

As to dependent claim 19, Reasor et al. teaches:

The computer-accessible medium of claim 18 wherein the markup language is an extensible markup language ("XML") (see e.g., para. [0017]).

As to dependent claim 21, Reasor et al. teaches:

The computer-accessible medium of claim 15, wherein the hardware entities include at least one of a motherboard (see e.g., para. [0014], line 18) and an add-in card of the computing device (see e.g., para. [0012], lines 11 – 13; i.e., add-in cards corresponds to CPU, memory, and any peripheral devices that may be externally connected).

Claims 1,2, 4 – 6, 9, 10, 14, and 16 are rejected under 35 U.S.C. 103(a) as being unpatentable over Reasor et al. (Pub. No. 2004/0083196) in view of Crisan et al. (Publication No. 2003/0172372) and further in view of Ibanez et al. (Publication No. 2004/0254978).

As to independent claim 1, Reasor et al teaches a method, comprising: building a central repository of data structures (see e.g., para. [0012], lines 19 – 21; i.e., the central repository corresponds to the database, in which the data structures corresponds to the information associated with hardware devices of a computing system), the data structures provided to the central repository by hardware entities of a computing device (see e.g., para. [0012], lines 14 – 22; i.e., the hardware entities corresponds to peripheral devices that may be externally connected, such as CPU, and memories, in which the system firmware identifies and queries connected hardware devices to determine the hardware properties for storing in the database); displaying hardware configuration settings of the hardware entities (see e.g., para. [0023] – para. [0024]; i.e., the hardware configuration settings of the hardware entities corresponds to hardware properties displayed in a browser, wherein the user is able to configure the hardware entities by adding a "Size" 400 to the XML hierarchy tree. It is interpreted that hardware configuration settings, such as "Address", "Attributes", and "Size" depicted in Fig. 4, are reconfigurable, wherein Reasor et al. explicitly mentions the addition of the attribute "Size" 400 may be added to memory region 310 and 311 of the XML tree. Furthermore, the XML representation uses a hypertext transfer protocol (HTTP) directly from the firmware to display the XML representation on a browser), and the hardware configuration settings based at least in part on the data structures provided to the central repository (see e.g., para. [0013]; i.e., the configurable properties of the hardware device are configurable using a markup language compatible browser, in which the properties are provided by the database). Reasor et al. does not specifically mention a pre-boot runtime of the computing device and a remote console communicatively coupled to the computing device via a network. Crisan et al. teaches a pre-boot runtime of the computing

device (see e.g., para. [0007]; i.e., the "system ROM" is a storage area of information used to initialize the basic input/output system "BIOS" of a computing device, wherein the powering-up of a computing device will execute the low-level functions, such as the BIOS and validity of peripheral devices discussed by Crisan et al., and further initialize the power on self test (POST). POST, is well known to one of ordinary skill in the art for querying and polling peripheral devices during system initialization, prior (emphasis added) to the initialization of the operating system (OS), wherein POST routines test various peripheral devices connected to the computing device in order to properly setup the utilization of the peripheral devices), a remote console communicatively coupled to the computing device via a network (see e.g., para. [0012]; i.e., the client computers are connected to the ROM server through the Internet) and updating hardware configuration settings of the hardware entities of the computing device (see e.g., para. [0011]). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate building a central repository of data structures provided by the hardware entities of a computing device of Reasor et al. with a pre-boot runtime of the computing device, and a remote console communicatively coupled to the computing device via a network and updating hardware configuration settings of the hardware entities of the computing device of Crisan et al. because the ROM that stores data structures of hardware devices and the basic startup code for executing low-level functions of a system can be updated during initialization of the computing device (see e.g., para. [0011]).

Both Reasor et al. and Crisan et al. do not specifically mention displaying hardware configuration settings of hardware entities using a browser running on a remote console communicatively coupled to the computing device via a network. Ibanez et al. teaches displaying

hardware configuration settings of hardware entities using a browser (see e.g., para. [0009] and para. [0031]; i.e., the interaction between the computing device and a remote computer system occurs through a Web browser, wherein the Web browser is able to configure hardware device self-installing software packages and their updates) running on a remote console communicatively coupled to the computing device via a network (see e.g., Fig. 1 and para. [0009]; i.e., network 102 is used to connect server 104 and client computing devices 108, 110, and 112). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate building a central repository of data structures provided by the hardware entities of a computing device of Reasor et al. as modified by a pre-boot runtime of the computing device, and a remote console communicatively coupled to the computing device via a network and updating hardware configuration settings of the hardware entities of the computing device of Crisan et al. with displaying hardware configuration settings of hardware entities using a browser running on a remote console communicatively coupled to the computing device via a network of Ibanez et al. because the Wake-On-Lan technology can remotely power-on a computing system and initiate maintenance over a network, wherein remotely powering-on a computing device without an operating system can be remedied by a pre-boot execution environment (PXE) (see e.g., para. [0009]).

As to dependent claim 2, Reasor et al. teaches:

The method of claim 1, further comprising changing at least one of the hardware configuration settings (see e.g., para. [0023]; i.e., the changing of at least one hardware configuration setting corresponds to adding the property "Size" 400 to one of the hardware's described in Fig. 1) in

response to input received via the browser (browser – see e.g., Fig. 2 and para. [0024]; i.e., Fig. 2 is a markup language compatible browser that allows hardware property configuration).

As to dependent claim 4, Reasor et al. teaches:

The method of claim 1 wherein displaying further includes displaying hardware configuration settings (see e.g., Fig. 2 and para. [0019] – [0020]; i.e., Fig. 2 corresponds to a device tree displayed on a device interface used to configure settings, such as adding a "Size" 400 property shown in Fig. 4) based at least in part on the data structures (see e.g., para. [0019]; i.e., the data structures corresponds to the database that is used to store the hardware properties, which in turn is used to construct the device tree 250) and nonvolatile data associated with the hardware entities (see e.g., "Microsoft Computer Dictionary 5th edition" and para. [0011]; i.e., non-volatile memory is defined as "A storage system that does not lose data when power is removed from it. Intended to refer to core memory, ROM, EPROM…", in which the functionality of hardware configuration comprises system software code stored on read-only-memory (ROM) or solid-state-memory).

As to dependent claim 5, Reasor et al. teaches:

The method of claim 2 wherein building the central repository further includes providing the central repository with the data structures (see e.g., para. [0013]; i.e., building the central repository and providing data structures corresponds to holding information about the hardware device in RAM as they are discovered, in the form of a tree format using descriptive properties and attributes that can be converted into XML) being described using a language convertible to a markup language (see e.g., para. [0017], lines 16 – 18; i.e., the tree format data structure

attributes and properties used to build the central repository is able to be parsed to display a hierarchal tree using an Extensible Markup Language (XML) browser, such as Fig. 2).

As to dependent claim 6, Reasor et al. teaches:

The method of claim 5 wherein the markup language is an extensible markup language ("XML") (see e.g., para. [0017]).

As to dependent claim 9, Reasor et al. teaches:

The method of claim 1 wherein the hardware entities include at least one of a motherboard (see e.g., para. [0014], line 18) and an add-in card of the computing device (see e.g., para. [0012], lines 11 – 13; i.e., add-in cards corresponds to CPU, memory, and any peripheral devices that may be externally connected).

As to dependent claim 10, Reasor et al. teaches:

The method of claim 1 wherein displaying hardware configuration settings (see e.g., Fig. 2 and para. [0019] – [0020]; i.e., Fig. 2 corresponds to a device tree displayed on a device interface used to configure settings, such as adding a "Size" 400 property shown in Fig. 4) includes displaying policy settings of the hardware entities of the computing device using the browser (see e.g., para. [0020]; each CPU has several properties, such as frequency, model number, and the like, which corresponds to policy settings displayed on an XML compatible browser), the policy settings based at least in part on the data structures provided to the central repository (see e.g., para. [0020]; i.e., the properties are a result of data structures stored in each of CPU 100, 101, 102, and 103).

As to dependent claim 14, Reasor et al. and Crisan et al. does not specifically mention the browser is a web browser executing on a remote console communicatively coupled to the

computing device via a network. Ibanez et al. teaches the browser is a web browser(see e.g., para. [0009] and para. [0031]; i.e., the interaction between the computing device and a remote computer system occurs through a Web browser, wherein the Web browser is able to configure hardware device self-installing software packages and their updates) running on a remote console communicatively coupled to the computing device via a network (see e.g., Fig. 1 and para. [0009]; i.e., network 102 is used to connect server 104 and client computing devices 108, 110, and 112). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate building a central repository of data structures provided by the hardware entities of a computing device of Reasor et al. as modified by a pre-boot runtime of the computing device, and a remote console communicatively coupled to the computing device via a network and updating hardware configuration settings of the hardware entities of the computing device of Crisan et al. with displaying hardware configuration settings of hardware entities using a browser running on a remote console communicatively coupled to the computing device via a network of Ibanez et al. because the Wake-On-Lan technology can remotely power-on a computing system and initiate maintenance over a network, wherein remotely powering-on a computing device without an operating system can be remedied by a pre-boot execution environment (PXE) (see e.g., para. [0009]).

As to dependent claim 16, Reasor et al. teaches a computer-accessible medium (ROM – see e.g., para. [0012]), wherein the instructions (software program – see e.g., para. [0012]) for generating the browser page (see e.g., Fig. 2 and para. [0024], lines 4 – 9; i.e., the browser is capable of displaying configurable hardware properties) further include instructions (software program – see e.g., para. [0012]), generating a browser page to be displayed in a web browser

(see e.g., para. [0023] – para. [0024]; i.e., the hardware configuration settings of the hardware entities corresponds to hardware properties displayed in a browser, wherein the user is able to configure the hardware entities by adding a "Size" 400 to the XML hierarchy tree. It is interpreted that hardware configuration settings, such as "Address", "Attributes", and "Size" depicted in Fig. 4, are reconfigurable, wherein Reasor et al. explicitly mentions the addition of the attribute "Size" 400 may be added to memory region 310 and 311 of the XML tree. Furthermore, the XML representation uses a hypertext transfer protocol (HTTP) directly from the firmware to display the XML representation on a browser). Both Reasor et al. and Crisan et al. do not specifically mention generating the browser page to be displayed in a web browser of a remote console communicatively coupled to the computing device via a network. Ibanez et al. teaches generating the browser page to be displayed in a web browser (see e.g., para. [0009] and para. [0031]; i.e., the interaction between the computing device and a remote computer system occurs through a Web browser, wherein the Web browser is able to configure hardware device self-installing software packages and their updates) of a remote console communicatively coupled to the computing device via a network (see e.g., Fig. 1 and para. [0009]; i.e., network 102 is used to connect server 104 and client computing devices 108, 110, and 112). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate a computer-accessible medium for generating a browser page of Reasor et al. as modified by the pre-boot runtime of the computing device of Crisan et al. with generating the browser page to be displayed in a web browser of a remote console communicatively coupled to the computing device via a network of Ibanez et al. because the Wake-On-Lan technology can remotely power-on a computing system and initiate maintenance over a network, wherein

remotely powering-on a computing device without an operating system can be remedied by a pre-boot execution environment (PXE) (see e.g., para. [0009]).

Claims 23 – 27, 29, and 30 are rejected under 35 U.S.C. 103(a) as being unpatentable over Reasor et al. (Pub. No. 2004/0083196) in view of Ibanez et al. (Publication No. 2004/0254978) and further in view of Mahmoud et al. (Patent No. 7,007,158).

As to independent claim 23, Reasor et al. teaches a computing device (computing system – see e.g., para. [0011]) comprising a processor (processor – see e.g., para. [0012], line 25) multiple hardware entities (see e.g., para. [0012], lines 11 – 13; i.e., the multiple hardware entities corresponds to any physical parts of the computing system, including the CPU, memory, and any peripheral devices) communicatively coupled to the processor (see e.g., para. [0012], lines 11 – 16; the hardware entities are communicatively connected to the processor in order for the firmware to query the hardware devices), nonvolatile memory coupled to the processor (ROM – see e.g., para. [0012], line 7; i.e., the nonvolatile memory stores the firmware and is executed by the processor to query the hardware entities), data structure corresponding to multiple hardware entities (see e.g., para. [0012], lines 14 – 22; i.e., the hardware entities corresponds to peripheral devices that may be externally connected, such as CPU, and memories, in which the system firmware identifies and queries connected hardware devices to determine the hardware properties for storing in the database), wherein the browser uses the data structures to display configuration settings (see e.g., Fig. 2 and para. [0024], lines 4 – 9; i.e., the browser is capable of displaying configurable hardware properties), the browser comprising a markup language (see e.g., para. [0017], lines 16 – 18; i.e., the tree format data structure attributes and

properties used to build the central repository is able to be parsed to display a hierarchal tree using an Extensible Markup Language (XML) browser, such as Fig. 2). Reasor et al. does not specifically mention a translator for converting the data structure to generate a browser page to display hardware configuration settings and monitoring a port of the computing device. Ibanez et al. teaches monitoring a port of the computing device. (see e.g., para. [0004] – para. [0005]; i.e., the computing devices input/output devices correspond to a network interface card (NIC), wherein the system continuously monitors for Wake-On-Lan (WOL) packets. Once the NIC has detected WOL packets transmitted by a remote computer, the computing device can be remotely turned on for remote maintenance). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the time the invention was made to incorporate the processor, multiple hardware entities, nonvolatile memory, data structures corresponding to multiple hardware entities, and a markup language browser for displaying hardware configuration settings of Reasor et al. with monitoring a port of the communication device of Ibanez et al. because the Wake-On-Lan technology can remotely power-on a computing system and initiate maintenance over a network, wherein remotely powering-on a computing device without an operating system can be remedied by a pre-boot execution environment (PXE) (see e.g., para. [0009]).

Both Reasor et al. and Ibanez et al. do not specifically mention a translator for converting data structure to generate a browser page to display hardware configuration settings. Mahmoud et al. teaches a translator for converting data structure to generate a browser page to display hardware configuration settings (see e.g., col. 2, lines 39 – 50; i.e., the data structure is passed to the BIOS of the storage handling controller for converting the data structure into an XML page

based on the data structure). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the processor, multiple hardware entities, nonvolatile memory, data structures corresponding to multiple hardware entities, and a markup language browser for displaying hardware configuration settings of Reasor et al. as modified by monitoring a port of the communication device of Ibanez et al. with the translator converting data structures corresponding to multiple hardware entities of Mahmoud et al. because the GUI can be presented to the user for easier configuration of the storage handling controller without the need of creating hardware specific graphic libraries (see e.g., col. 2, lines 66 – 67, and col. 3, lines 1 – 9).

As to dependent claim 24, Reasor et al. does not specifically mention a network communicatively coupled to the computing device, a remote console communicatively coupled to the network, a translator to update nonvolatile data associated with at least one of the hardware entities in response to markup language data received from the browser. Ibanez et al. teaches a remote console communicatively coupled to a network, and the browser to execute on the remote console (see e.g., para. [0009] and para. [0031]; i.e., the interaction between the computing device and a remote computer system occurs through a Web browser, wherein the Web browser is able to configure hardware device self-installing software packages and their updates). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the computing device of Reasor et al. with a remote console communicatively coupled to a network, and the browser to execute on the remote console of Ibanez et al. because the Wake-On-Lan technology can remotely power-on a computing system and initiate maintenance over a network, wherein remotely powering-on a computing device

without an operating system can be remedied by a pre-boot execution environment (PXE) (see e.g., para. [0009]).

Both Reasor et al. and Ibanez et al do not specifically mention the translator to update nonvolatile data associated with at least one of the hardware entities in response to markup language data received from the browser. Mahmoud et al. teaches a translator to update nonvolatile data associated with at least one of the hardware entities in response to markup language data received from the browser (see e.g., col. 11, lines 7 – 20; i.e., user selections and commands received from the XML browser are transmitted back to the firmware for updating the nonvolatile data). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the changing of at least one hardware configuration settings of Reasor et al. as modified by a remote console communicatively coupled to a network, and the browser to execute on the remote console of Ibanez et al. with the translator and updating nonvolatile data using the translator of Mahmoud et al. because the XML based storage handling controller can be configured remotely using XML capable software (see e.g., col. 11, lines 17 – 20).

As to dependent claim 25, both Reasor et al. and Mahmoud et al. do not specifically mention the browser is a web browser. Ibanez et al. teaches the browser is a web browser (see e.g., para. [0009] and para. [0031]; i.e., the interaction between the computing device and a remote computer system occurs through a Web browser, wherein the Web browser is able to configure hardware device self-installing software packages and their updates).

As to dependent claim 26, Reasor et al. teaches a processor (processor –see e.g., para. [0012]), a computer-accessible medium (ROM – see e.g., para. [0012]), executable instruction

instructions (software program – see e.g., para. [0012]), and executing a browser on a computing device (see e.g., para. [0024]; lines 1 – 9), but does not teach a translator to update nonvolatile data associated with at least one of the hardware entities in response to markup language data received from the browser. Ibanez et al. teaches monitoring a port of the computing device. (see e.g., para. [0004] – para. [0005]; i.e., the computing devices input/output devices correspond to a network interface card (NIC), wherein the system continuously monitors for Wake-On-Lan (WOL) packets. Once the NIC has detected WOL packets transmitted by a remote computer, the computing device can be remotely turned on for remote maintenance). Mahmoud et al. teaches a translator (see e.g., col. 2, lines 39 – 50; i.e., the data structure is passed to the BIOS of the storage handling controller for converting the data structure into an XML page based on the data structure) to update nonvolatile data associated with at least one of the hardware entities in response to markup language data received from the browser (see e.g., col. 11, lines 7 – 20; i.e., user selections and commands received from the XML browser are transmitted back to the firmware for updating the nonvolatile data). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the changing of at least one hardware configuration settings of Reasor et al. as modified by monitoring a port of the communication device of Ibanez et al. with the translator and updating nonvolatile data using the translator of Mahmoud et al. because the XML based storage handling controller can be configured remotely using XML capable software (see e.g., col. 11, lines 17 – 20).

As to dependent claim 27, Reasor et al. teaches hardware entities (see e.g., para. [0012], lines 11 – 13) wherein the hardware entities include firmware units (firmware – see e.g., para. [0012], lines 1 – 8) having data structures stored therein (see e.g., para. [0012, lines 1 – 7; i.e.,

the data structures stored in the firmware corresponds to the basic software programs that can be accessed immediately when the computing device is powered-on), and the data structures contributing to the central repository (see e.g., para. [0012], lines 8 – 21; the data structures associated with the firmware contributes to the database by identifying connected hardware devices and sending the information to the database). Both Reasor et al. and Ibanez et al. do not specifically mention the data structures stored in the firmware unit are in the form of binaries, wherein the binaries contribute as the data structure for the central repository. Mahmoud et al. teaches a firmware unit storing data structures in the form of binaries (see e.g., col. 1, lines 21 – 38 and col. 5, lines 37 – 49; i.e., the storage handling firmware stores byte-codes, in which byte-code corresponds to binaries) wherein the binaries contribute to the central repository (see e.g., col. 6, lines 4 – 12). Therefore, it would have been obvious to one of ordinary skill at the time the invention was made to incorporate the hardware entities including firmware units and data structures contributing to the central repository of Reasor et al. as modified by monitoring a port of the communication device of Ibanez et al. with the firmware unit storing data structures in the form of binaries and contributing to the central repository of Mahmoud et al. because the just-in-time complier or interpreter provides transformation of the byte-codes into machine code for processing (see e.g., col. 5, lines 44 – 49).

As to dependent claim 29, Reasor et al. teaches a nonvolatile memory unit (ROM – see e.g., para. [0012]) comprising a firmware unit (firmware – see e.g., para. [0012], lines 1 – 8) of a motherboard (motherboard – see e.g., para. [0014]; cells corresponds to motherboards in the computing device) of a computing system (computer – see e.g., para. [0012]).

As to dependent claim 30, Reasor et al. teaches the nonvolatile memory unit (ROM – see e.g., para. [0011]) of the computing device (computer – see e.g., para. [0012]).

Claims 7, 8, 11, 16, 20, 28, and 31 are rejected under 35 U.S.C. 103(a) as being unpatentable over Reasor et al. (Pub. No. 2004/0083196) in view of Crisan et al. (Publication No. 2003/0172372) in view of Ibanez et al. (Publication No. 2004/0254978) and further in view of Mahmoud et al. (Patent No. 7,007,158).

As to dependent claim 7, Reasor et al., Crisan et al., and Ibanez et al. do not specifically mention executing a translator on a computing device, wherein the translator converts the data structures into XML prior to displaying the hardware configuration settings using the browser. Mahmoud et al. teaches a translator for translating data structures into XML (see e.g., col. 2, lines 39 – 50; i.e., the data structure is passed to the BIOS of the storage handling controller for converting the data structure into an XML page based on the data structure) prior to displaying the hardware configuration setting using the browser (see e.g., col. 2, lines 39 – 50; i.e., the data structure must be passed to the BIOS before the BIOS can construct an XML page). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the building of a central repository populated by data structures provided by hardware entities, and displaying hardware configuration settings using a XML compatible browser of Reasor et al. as modified by a pre-boot runtime of the computing device, and a remote console communicatively coupled to the computing device via a network and updating hardware configuration settings of the hardware entities of the computing device of Crisan et al. as further modified by displaying hardware configuration settings of hardware entities using a

browser running on a remote console communicatively coupled to the computing device via a network of Ibanez et al. with the translator and converting the data structure prior to displaying the browser of Mahmoud et al. because the GUI can be presented to the user for easier configuration of the storage handling controller without the need of creating hardware specific graphic libraries (see e.g., col. 2, lines 66 – 67, and col. 3, lines 1 – 9).

As to dependent claim 8, Reasor et al. teaches changing at least one of the hardware configuration settings (see e.g., para. [0023]; i.e., the changing of at least one hardware configuration setting corresponds to adding the property "Size" 400 to one of the hardware's described in Fig. 1), and updating the nonvolatile data associated with the hardware entities with XML data received from the browser (see e.g., [0027], lines 25 – 30). Reasor et al., Crisan et al., and Ibanez et al. do not specifically mention a translator being executed on the computing device, and updating nonvolatile data associated with the hardware entities with XML data received from the browser. Mahmoud et al. teaches a translator (see e.g., col. 2, lines 39 – 50; i.e., the data structure is passed to the BIOS of the storage handling controller for converting the data structure into an XML page based on the data structure), and wherein the translator is also able to update nonvolatile data associated with the hardware entities with XML data received from the browser (see e.g., col. 11, lines 7 – 20; i.e., user selections and commands received from the XML browser are transmitted back to the firmware for updating the nonvolatile data). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the changing of at least one hardware configuration settings of Reasor et al. as modified by a pre-boot runtime of the computing device, and a remote console communicatively coupled to the computing device via a network and updating hardware

configuration settings of the hardware entities of the computing device of Crisan et al. as further modified by displaying hardware configuration settings of hardware entities using a browser running on a remote console communicatively coupled to the computing device via a network of Ibanez et al. with the translator and updating nonvolatile data using the translator of Mahmoud et al. because the XML based storage handling controller can be configured remotely using XML capable software (see e.g., col. 11, lines 17 – 20).

As to dependent claim 11, Reasor et al. teaches building the central repository (see e.g., para. [0012], lines 19 – 21; i.e., the central repository corresponds to the database) of the data structures (see e.g., para. [0012], lines 16 – 21; i.e., the data structures corresponds to information regarding configurable hardware properties, which is reported back to the computing system for acknowledgement of connected hardware devices) includes building the central repository in a system memory (see e.g., para. [0013], lines 1- 2; i.e., the system memory corresponds to filling the database with hardware information residing in RAM) of the computing device (see e.g., para. [0012]), the data structures being stored in option read only memories ("ROMs") of the hardware entities (see e.g., para. [0012], lines 4 – 8; i.e., firmware is responsible for collecting hardware device information, in which the firmware is code stored in ROM), the central repository being built (see e.g., para. [0012], lines 1 – 4 and para. [0020]; i.e., firmware for the computing system is immediately accessed when the computing system is turned on, wherein the firmware identifies hardware by polling each hardware attached to the computing system). Crisan et al. teaches a pre-boot runtime of the computing device (see e.g., para. [0007]; i.e., the "system ROM" is a storage area of information used to initialize the basic input/output system "BIOS" of a computing device, wherein the powering-up of a computing

device will execute the low-level functions, such as the BIOS and validity of peripheral devices discussed by Crisan et al., and further initialize the power on self test (POST). POST, is well known to one of ordinary skill in the art for querying and polling peripheral devices during system initialization, prior (emphasis added) to the initialization of the operating system (OS), wherein POST routines test various peripheral devices connected to the computing device in order to properly setup the utilization of the peripheral devices). Reasor et al., Crisan et al., and Ibanez et al. do not specifically mention the data structures obtained from binaries being stored in memory. Mahmoud et al. teaches data structures obtained from binaries stored in memory (see e.g., col. 1, lines 21 – 38 and col. 5, lines 37 – 49; i.e., the storage handling firmware stores byte-codes, in which byte-code corresponds to binaries). Therefore, it would have been obvious to one of ordinary skill at the time the invention was made to incorporate the hardware entities including firmware units and data structures contributing to the central repository of Reasor et al. as modified by a pre-boot runtime of the computing device, and a remote console communicatively coupled to the computing device via a network and updating hardware configuration settings of the hardware entities of the computing device of Crisan et al. as further modified by displaying hardware configuration settings of hardware entities using a browser running on a remote console communicatively coupled to the computing device via a network of Ibanez et al. with the firmware unit storing data structures in the form of binaries and contributing to the central repository of Mahmoud et al. because the just-in-time complier or interpreter provides transformation of the byte-codes into machine code for processing (see e.g., col. 5, lines 44 – 49).

As to dependent claim 20, Reasor et al teaches the instructions (ROM – see e.g., para. [0012]) for generating a browser page (see e.g., Fig. 2 and para. [0024], lines 4 – 9; i.e., the browser is capable of displaying configurable hardware properties). Reasor et al., Crisan et al. and Ibanez et al. does not specifically mention instructions to execute a translator to convert the data structures into the XML prior to generating the browser page to display the hardware configuration settings. Mahmoud et al. teaches instructions to execute a translator to convert the data structures into the XML prior to generating the browser page to display the hardware configuration settings (see e.g., col. 2, lines 39 – 50; i.e., the data structure is passed to the BIOS of the storage handling controller for converting the data structure into an XML page based on the data structure). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the building of a central repository populated by data structures provided by hardware entities, and displaying hardware configuration settings using a XML compatible browser of Reasor et al. as modified by a pre-boot runtime of the computing device, and a remote console communicatively coupled to the computing device via a network and updating hardware configuration settings of the hardware entities of the computing device of Crisan et al. as further modified by displaying hardware configuration settings of hardware entities using a browser running on a remote console communicatively coupled to the computing device via a network of Ibanez et al. with the translator and converting the data structure prior to displaying the browser of Mahmoud et al. because the GUI can be presented to the user for easier configuration of the storage handling controller without the need of creating hardware specific graphic libraries (see e.g., col. 2, lines 66 – 67, and col. 3, lines 1 – 9).

As to dependent claim 28, Reasor et al. teaches contributing data structures to the central repository (see e.g., para. [0012], lines 1 – 4; i.e., firmware for the computing system is immediately accessed when the computing system is turned on) of the computing device (see e.g., para. [0012], lines 1 – 21; i.e., the system firmware identifies connected hardware and queries the hardware devices to determine the properties in order to construct a database). Ibanez et al. teaches monitoring a port of the computing device. (see e.g., para. [0004] – para. [0005]; i.e., the computing devices input/output devices correspond to a network interface card (NIC), wherein the system continuously monitors for Wake-On-Lan (WOL) packets. Once the NIC has detected WOL packets transmitted by a remote computer, the computing device can be remotely turned on for remote maintenance). Both Reasor et al. and Ibanez et al. do not specifically mention pre-boot runtime of the computing device and binaries contributing as data structures to the central repository. Crisan et al. teaches pre-boot runtime of the computing device (see e.g., para. [0007]; i.e., the "system ROM" is a storage area of information used to initialize the basic input/output system "BIOS" of a computing device, wherein the powering-up of a computing device will execute the low-level functions, such as the BIOS and validity of peripheral devices discussed by Crisan et al., and further initialize the power on self test (POST). POST, is well known to one of ordinary skill in the art for querying and polling peripheral devices during system initialization, prior (emphasis added) to the initialization of the operating system (OS), wherein POST routines test various peripheral devices connected to the computing device in order to properly setup the utilization of the peripheral devices). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate contributing data structures to the central repository of the computing device as modified by

monitoring a port of the computing device of Ibanez et al. with pre-boot runtime of the computing device of Crisan et al. because the ROM that stores data structures of hardware devices and the basic startup code for executing low-level functions of a system can be updated during initialization of the computing device (see e.g., para. [0011]).

Reasor et al., Crisan et al., and Ibenaz et al. do not specifically mention binaries contributing as data structures to the central repository. Mahmoud et al. teaches binaries contributing as data structures to the central repository (see e.g., col. 6, lines 4 – 12). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate contributing data structures to the central repository of the computing device as modified by monitoring a port of the computing device of Ibanez et al. as further modified by pre-boot runtime of the computing device of Crisan et al. with the firmware unit storing data structures in the form of binaries and contributing to the central repository of Mahmoud et al. because the just-in-time complier or interpreter provides transformation of the byte-codes into machine code for processing (see e.g., col. 5, lines 44 – 49).

As to dependent claim 31, Reasor et al. does not specifically mention the translator is configured to monitor the port of the computing device during pre-boot runtime of the computing device prior to loading an operating system and is further configured to generate the browser page in response to the request during the pre-boot runtime. Crisan et al. teaches configuring hardware entities (see e.g., para. [0007]) during pre-boot runtime of the computing device (see e.g., para. [0007]; i.e., the "system ROM" is a storage area of information used to initialize the basic input/output system "BIOS" of a computing device, wherein the powering-up of a computing device will execute the low-level functions, such as the BIOS and validity of

peripheral devices discussed by Crisan et al., and further initialize the power on self test (POST). POST, is well known to one of ordinary skill in the art for querying and polling peripheral devices during system initialization, prior (emphasis added) to the initialization of the operating system (OS), wherein POST routines test various peripheral devices connected to the computing device in order to properly setup the utilization of the peripheral devices), a remote console communicatively coupled to the computing device via a network (see e.g., para. [0012]; i.e., the client computers are connected to the ROM server through the Internet) and updating hardware configuration settings of the hardware entities of the computing device (see e.g., para. [0011]) prior to loading an operating system (see e.g., para. [0007]; i.e., power on self test (POST) and BIOS are initialized before the loading of the operating system, wherein the BIOS is further executable code used to handle low level input/output transactions prior to the operating system being loaded). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate building a central repository of data structures provided by the hardware entities of a computing device of Reasor et al. with configuring hardware entities during a pre-boot runtime of the computing device prior to loading an operating system of Crisan et al. because the ROM that stores data structures of hardware devices and the basic startup code for executing low-level functions of a system can be updated during initialization of the computing device (see e.g., para. [0011]).

Both Reasor et al. and Crisan et al. does not specifically mention monitoring ports of a computing device. Ibanez et al. teaches generating a browser page (see e.g., para. [0009] and para. [0031]; i.e., the interaction between the computing device and a remote computer system occurs through a Web browser, wherein the Web browser is able to configure hardware device

self-installing software packages and their updates), and monitoring a port of the computing device. (see e.g., para. [0004] – para. [0005]; i.e., the computing devices input/output devices correspond to a network interface card (NIC), wherein the system continuously monitors for Wake-On-Lan (WOL) packets. Once the NIC has detected WOL packets transmitted by a remote computer, the computing device can be remotely turned on for remote maintenance). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the time the invention was made to incorporate the processor, multiple hardware entities, nonvolatile memory, data structures corresponding to multiple hardware entities, and a markup language browser for displaying hardware configuration settings of Reasor et al. as modified by configuring hardware entities during a pre-boot runtime of the computing device prior to loading an operating system of Crisan et al. with generating a browser page and monitoring a port of the communication device of Ibanez et al. because the Wake-On-Lan technology can remotely power-on a computing system and initiate maintenance over a network, wherein remotely powering-on a computing device without an operating system can be remedied by a pre-boot execution environment (PXE) (see e.g., para. [0009]).

Reasor et al., Crisan et al., and Ibanez et al. do not specifically mention a translator. Mahmoud et al. teaches a translator (see e.g., col. 11, lines 7 – 20; i.e., user selections and commands received from the XML browser are transmitted back to the firmware for updating the nonvolatile data). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the time the invention was made to incorporate the processor, multiple hardware entities, nonvolatile memory, data structures corresponding to multiple hardware entities, and a markup language browser for displaying hardware

configuration settings of Reasor et al. as modified by configuring hardware entities during a pre-boot runtime of the computing device prior to loading an operating system of Crisan et al. as further modified by monitoring a port of the communication device of Ibanez et al. with the translator of Mahmoud et al. because the XML based storage handling controller can be configured remotely using XML capable software (see e.g., col. 11, lines 17 – 20).

Inquiries

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Henry Vuu whose telephone number is (571) 270-1048. The examiner can normally be reached on 8-5.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Weilun Lo can be reached on (571) 272-4847. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

Henry Vuu

4/16/2007

BA HUYNH
PRIMARY EXAMINER